Une heuristique constructive guidée par relaxations de problèmes d'ordonnancement

S. Boivin, B. Pralong, A. Zanarini, et G. Pesant

Département de génie informatique, École Polytechnique de Montréal, C.P. 6079, succ. Centre-ville, Montréal Canada, H3C 3A7

1 Introduction

Ce document décrit notre approche de résolution du Problème de planification des techniciens et des interventions pour les Télécommunications chez France Télécom dans le cadre du "Challenge" ROADEF 2007. Notre première décision fut d'utiliser certains des logiciels mis à notre disposition par la société ILOG afin de profiter de leur efficacité et de leur puissance expressive pour la résolution de problèmes combinatoires. Ceci permettait également de réduire nos efforts d'implémentation.

Notre approche s'appuie sur le fait que, une fois les équipes de techniciens formées, nous nous retrouvons face à un problème d'ordonnancement de tâches assez classique. Évidemment, la formation optimale de ces équipes est cependant très difficile. Nous avons donc conçu une heuristique constructive formant des équipes jour après jour et planifiant les interventions avec le logiciel ILOG Scheduler. Ces planifications sont en fait lancées tout au long de la construction d'une solution afin de guider le processus heuristique.

Notre algorithme procède donc comme suit. Pour chaque jour, en commençant par le premier et jusqu'à ce que toutes les interventions aient été planifiées, plusieurs essais de formation d'équipes sont entrepris. Le nombre de ces essais est fixé en fonction du temps de calcul résiduel et d'une approximation du nombre de jours restant à planifier. Chaque essai est évalué en résolvant une relaxation du problème de planification, décrite à la prochaine section. Le meilleur essai de formation d'équipe est définitivement adopté pour ce jour et nous passons ensuite au jour suivant.

2 Modélisation en un problème d'ordonnancement

Comme nous l'avons indiqué en introduction, notre heuristique constructive est guidée à chaque étape par la résolution d'un problème d'ordonnancement. Cette section décrit la manière dont nous avons modélisé ce problème.

Le Problème d'ordonnancement consiste à allouer des ressources limitées à des activités dans le temps. La traduction du problème qui nous intéresse dans ce formalisme est assez directe : nos ressources sont les équipes et nos activités, les interventions. Il y a cependant plusieurs aspects du problème qui ne sont pas aussi immédiats à modéliser.

2.1 Activités

En plus de créer une activité de durée correspondante pour chaque intervention, nous ajoutons une activité de durée nulle pour chaque niveau de priorité. L'activité supplémentaire du niveau de priorité p est ensuite contrainte à débuter au plus tôt après chacune des activités de priorité p. Cet artifice facilite l'expression de la fonction de coût comme une somme pondérée du moment où débute l'activité représentant chacun des niveaux de priorité.

Les contraintes de préséance entre interventions sont facilement exprimées avec les variables du moment d'exécution des activités correspondantes et automatiquement prises en compte par ILOG Scheduler.

2.2 Ressources

Le problème d'ordonnancement que nous résolvons à l'étape j sert à choisir les équipes que nous formerons au jour j. Nous connaissons alors les équipes des $j-1$ premiers jours mais pas

encore les autres. Afin de guider le choix heuristique des équipes du jour j, nous résolvons une relaxation du problème dans laquelle les ressources disponibles à partir du jour $j+1$ sont deux relaxations d'équipes. La première s'appuie simplement sur le nombre de techniciens disponibles un jour donné. Une ressource discrète est définie et sa capacité est ajustée pour correspondre au nombre de techniciens disponibles au fil des jours. La seconde est en fait une famille de ressources : pour chaque domaine d et chaque niveau n, nous définissons une ressource discrète dont la capacité varie selon le jour et qui correspond au nombre de techniciens disponibles ce jour-là qui possèdent une compétence d'au moins n pour le domaine d. En exigeant que toute activité consomme de chacune de ces ressources une quantité correspondante aux qualifications requises pour effectuer l'intervention, on évite de nombreux ordonnancements irréalisables au-delà du jour j et on obtient une meilleure borne supérieure sur le coût de la solution qui sera construite.

Telle quelle, cette formulation permet cependant qu'une activité soit planifiée en partie sur une journée et en partie sur la suivante, ce qui est clairement illégal dans notre contexte réel. Afin d'interdire qu'une telle situation se produise, nous avons exploité la possibilité en ILOG Scheduler de spécifier des intervalles périodiques pendant lesquels une ressource n'est plus disponible. Dans notre cas, ces intervalles de durée nulle coincident avec la fin de chaque journée.

Penchons-nous maintenant sur les ressources offertes les j premiers jours. Alors que dans un problème d'ordonnancement standard, les ressources sont disponibles pour la durée de la planification, dans notre cas les ressources que sont les équipes ne servent que lors d'un jour donné. Nous définissons donc pour chaque équipe une ressource disjonctive (de capacité 1) pour laquelle nous spécifions une capacité maximale de 0 tous les jours sauf celui où cette équipe est constituée. Chaque ressource-équipe représente une alternative pour une activité-intervention. Le concept de ressources alternatives est déjà prévu en ILOG Scheduler et permet de spécifier qu'une activité consommera une ressource parmi l'ensemble des équipes ayant les compétences nécessaires pour l'effectuer. Pour permettre à une activité de ne pas nécessairement choisir une équipe existante mais d'apparaître plus tard (après le jour j) dans la planification, une équipe artificielle de capacité illimitée à partir du jour $j+1$ est ajoutée à l'ensemble des alternatives. Comme pour les ressources précédentes, des intervalles périodiques de non disponibilité sont spécifiés.

Pour terminer, mentionnons que le modèle d'ordonnancement est modifié d'une résolution à une autre plutôt que redéfini à partir de zéro. Puisqu'il sera résolu plusieurs fois chaque jour de la planification, pour chaque essai de formation d'équipe, on évite ainsi un gaspillage de temps et d'espace mémoire.

2.3 Heuristiques de recherche d'ordonnancement

Puisque les heuristiques de recherche d'ordonnancement par défaut de ILOG Scheduler ne permettaient pas de résoudre nos exemplaires dans des temps raisonnables, nous les avons adapté à notre contexte. Lorsque des ressources alternatives sont présentes, comme c'est le cas ici, il est généralement recommandé de choisir d'abord une ressource pour chaque activité, puis de fixer le moment où on les exécute. Nous considérons les activités des plus prioritaires aux moins prioritaires et, parmi les activités de même priorité, celles considérées les plus difficiles d'abord, selon le critère décrit à la section 3. Une fois l'activité identifiée, nous considérons les ressources alternatives en favorisant celles qui apparaissent plus tôt dans la planification et qui correspondent à des équipes de plus petite taille (tout en étant suffisamment qualifiées, évidemment).

3 Heuristiques de création d'équipe

La procédure de création d'équipe débute par le choix d'un sous-ensemble de tâches à exécuter. Seules les tâches qui n'ont pas été affectées dans les jours précédant le jour courant sont considérées. Ces tâches sont sélectionnées selon une probabilité dépendante du niveau de priorité de la tâche ainsi que d'une approximation de son niveau de difficulté. Le *niveau de difficulté approximatif d'une tâche* est défini comme le maximum des niveaux de difficulté de ses domaines. On définit le niveau de difficulté d'un domaine d'une tâche comme la somme des techniciens requis pondérée en fonction des niveaux de compétences demandés.

Par la suite, on essaie d'affecter les techniciens disponibles aux tâches sélectionnées. Ce sous-problème est résolu par une approche de Programmation par contraintes (PPC). Dans le modèle

utilisé, chaque technicien est représenté par une variable dont le domaine est le sous-ensemble de tâches à réaliser. Pour chaque paire de domaine et niveau de compétence des tâches, on exprime ces exigences par une contrainte globale de cardinalité. Lorsqu'aucune affectation des techniciens aux tâches n'est trouvée, on supprime une tâche du sous-ensemble des tâches à couvrir et on relance la résolution par PPC sur le nouveau modèle. Cette procédure sera exécutée pendant un nombre pré-déterminé d'itérations.

Entre chacun des appels à la procédure de création d'équipes, la probabilité de sélection des tâches qui ont déjà été sélectionnées (lors d'une itération précédente) est diminuée dans le but de diversifier les équipes créées et évaluées.

Lorsque l'heuristique de création d'équipes pour le jour j ne donne aucun résultat, une heuristique de secours est appelé pour proposer une création d'équipes valide pour le jour j en fonction des interventions déjà planifiées. Le fonctionnement de l'heuristique est aussi basé sur l'ordre de priorité décrit plus haut. Cette heuristique travaille de la manière suivante. On choisit un ensemble d'interventions n'ayant pas été encore planifiées. Cet ensemble est choisi aléatoirement en utilisant un biais sur la priorité de l'intervention. Le nombre d'interventions choisies dépend aussi du nombre de techniciens disponibles et on veille à ce qu'on utilise pas plus de ressources qu'il n'y en a de disponible pour le jour j. Une fois que l'ensemble des ressources est choisi, on essaie de construire un ensemble de super-tâches, c'est-à-dire la réunion de plusieurs tâches entre elles, pouvant être effectué par une seule et même équipe. La création de cet ensemble de super-tâches se fait en utilisant un simple algorithme glouton. Dès lors que l'on a notre ensemble de super-tâches, il ne reste plus qu'à former des équipes. La formation des équipes se fait de manière simple en utilisant un algorithme glouton.

4 Abandon d'interventions

Les tâches ont été abandonnées en fonction de leur niveau de difficulté et du coût associé à l'abandon de celles-ci. Le niveau de difficulté d'une tâche est calculé en fonction du critère décrit à la section 3 ainsi qu'en fonction du temps d'exécution de la tâche. Ce niveau de difficulté est par la suite divisé par le coût associé à l'abandon de la tâche. On favorise donc les tâches les plus difficiles couvrant le plus grand horizon de l'horaire et de plus petit coût d'abandon. Les tâches sont ensuite triées en fonction de leur potentiel d'abandon. On utilise un algorithme glouton de résolution du problème de sac-à-dos. La capacité du sac est le budget alloué à l'abandon et chacune des tâches représente un item de poids égal au coût d'abandon de cette tâche et de valeur égale au potentiel d'abandon. Il est à noter que seulement les tâches qui ne figurent pas dans la liste de prédécesseurs des autres tâches peuvent être abandonnées.

5 Résultats

Cette section présente les résultats que nous avons obtenus pour les exemplaires des ensembles A et B aux tableaux **??** et **??**. Puisqu'il y a une composante aléatoire à notre approche, nous rapportons la meilleure valeur et la valeur moyenne pour quelques exécutions.

Notons que les exemplaires 4, 5, 7 et 8 de l'ensemble B n'ont pu être résolus dans le temps accordé. Une explication possible est que le nombre de domaines et de niveaux de compétences de ces exemplaires se traduisent en un grand nombre de ressources alternatives créées pour notre modèle d'ordonnancement, freinant sa résolution.

6 Discussion

Nous pouvons dresser un premier bilan de notre expérience. Bien que les solutions produites paraissent intéressantes, particulièrerement pour les exemplaires de petite et de moyenne taille, les problèmes d'ordonnancement sont parfois assez longs à résoudre, ce qui induit des temps trop longs simplement pour construire une première solution. Il aurait évidemment été intéressant de mettre en place des heuristiques d'amélioration locale après une phase initiale de construction de solution mais notre choix de guider cette construction en résolvant une relaxation du problème entier semble trop coûteux en temps de calcul pour des exemplaires de grande taille, ne laissant plus de temps pour une phase de post-optimisation.

Tab. 1. Résultats - Ensemble A

Exemplaire	Meilleure	Moyenne
data1	2340	2340.0
data2	4755	4755.0
data3	14160	16245.0
data4	14760	14760.0
data5	37560	37762.5
data6	25200	29280.0
data7	32520	33412.5
data8	21480	22533.5
data9	32058	34144.0
data10	45215	47675.0

Tab. 2. Résultats - Ensemble B

Exemplaire	Meilleure	Moyenne
data1	49020	51183.75
data2	27540	32395.00
data3	27630	33393.75
data4	ND	ND
data5	ND	ND
data6	41880	41880.00
data7	ND	ND
data8	ND	ND
data9	43440	46477.50
data10	51870	55207.50